UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/604,987	06/28/2000	Srivatsan Parthasarathy	MS146910.1	6447

27195 7590 12/17/2003

AMIN & TUROCY, LLP
24TH FLOOR, NATIONAL CITY CENTER
1900 EAST NINTH STREET
CLEVELAND, OH 44114

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2124	8

DATE MAILED: 12/17/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PRG

Office Action Summary	Application No. 09/604,987	Applicant(s) PARTHASARATHY ET AL.	
	Examiner Tuan A Vu	Art Unit 2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time mey be available under the provisions of 37 CFR 1.136(a). In no event, however, mey e reply be timely filed after SIX (6) MONTHS from the meiling dete of this communicetion.
- If the period for reply specified ebove is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Feilure to reply within the set or extended period for reply will, by statuta, causa tha application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *15 October 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-35* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-35* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on *28 June 2000* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. §§ 119 and 120

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) ☐ The translation of the foreign language provisional application has been received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Oraftsperson's Patent Orawing Review (PTO-948)
3) ☐ Information Oisclosure Statement(s) (PTO-1449) Paper No(s) _____ .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 10/15/2003.

As indicated in Applicant's response, claims 27, 30 have been amended. Claims 1-35 are pending in the office action.

Claims objections

2. Claims 6 and 15 are objected to because of the following informalities: there appears to be an error in verb/subject accordance between the elements 'contents' and 'has been modified' as recited in line 2. Examiner will see this as being "contents ... have been modified" to examine the merits of the claims. Appropriate correction is required.

3. Claim 24 is objected to because the element recited as 'the hash of the at least one module' preceding element 'an actual hash value of the at least one module' may seem equivocal; and should be corrected to be "[the] –said-- hash of the at least one module" to make it clear that the hash referred to is the hash from the manifest recited in the base claim in order to effect the comparison.

Allowable Subject Matter

4. The previously indicated allowability of claims 9, 19-2, 10-17, 22 is withdrawn in view of the newly discovered reference(s) to Renaud et al. USPN: 5,958,051. Rejections based on the newly cited reference(s) follow.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person

having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-8, 18, and 23-25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Renaud et al., USPN: 5,958,051 (hereinafter Renaud), in view of Buxton, USPN: 6,182,279 (hereinafter Buxton).

As per claim 1, Renaud discloses a method for integrity checking employable by application programs at runtime (e.g. Fig. 6; col. 9, line 40 to col. 10, line 37), such method comprising:

providing an assembly (e.g. *data structure 300* – Fig. 3a) that contains a list of versioned modules (e.g. Fig. 3a-b; *data files 304-314, version of the file* - col. 6, lines 46-64; col. 7, lines 31-36 – Note: Java class files or applets are equivalent to modules) that make up the assembly;

providing a manifest (e.g. *signature file 302* – Fig. 3b) with a hash value of data related to at least one module of the list of modules (e.g. *identifier, one-way hash* - col. 7, lines 15-27).

But Renaud does not specify that the hash of the list of modules making up assembly is a hash of the contents of at least one module. Renaud, however, teaches arranging modules or file data in such way that digital signature is combined with digest encryption algorithm and that the manifest can include data related to the modules comprising the assembly (e.g. *data relating to each data file, MD5, MD2* - col. 7, lines 24-54) to facilite processing and security checking. Buxton, in a system to load objects in an object-oriented user interface environment like Renaud's activating of applets and Java files via an GUI screen (Renaud: Fig. 5a), discloses using *templates* (e.g. *DLL storage 205, OLE Libraries, Template 208* - Fig. 2 – Note: Loading dynamic libraries and linking them to applications is equivalent to runtime linking) or file viewer, i.e. manifest, (*file viewer* - col. 9, lines 8-25) for associating components digital signature

for tracking of object identification (e.g. col. 12, lines 1-15; cols. 19-20) and integrity checking (e.g. Fig. 7-8B) and further discloses the hash of the contents of components making up *container* to be loaded via a template execution with each component being a self-contained module (e.g. Fig. 2; Fig. 3A; col. 9, lines 8-25; col. 8, lines 55-63). It would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the integrity checking of versioned modules listed in the manifest by Renaud such that such manifest lists not only the data modules hashed identifiers or data related to those modules but also a hash of the contents of such modules just as taught by Buxton. Renaud's hashing of module identifier differs from Buxton's module content hashing in that the former requires less resources, but in case processing resources are available so to enable encryption of the whole module content as in Buxton's system, one ordinary skill in the art would be motivated to do so is that submitting the modules as a whole via a digest encryption upon the time when the modules has been originally created would make it more accurate for subsequent integration checking as intended by the file download and identifier checking by Renaud.

As per claim 2, in view of the combined teachings of Renaud and Buxton from claim 1, the limitation of providing a hash of the contents each module that constitutes the assembly would have been obvious because the integrity checking would fulfill the same purpose as matching the integrity of such module at runtime, the difference being matching hashed contents would require more resources but would yield integrity of data on the data itself as mentioned in claim 1 above.

As per claim 3 and 4, Renaud further discloses providing identity information in the manifest (e.g. Fig. 3b, *additional data* – col. 6, lines 55-64) as well as publisher and version information.

As per claim 5, in reference to claim 1, Renaud discloses hash of the identifiers of versioned modules in the assembly and suggests placing a signature (or hashed representation)of the manifest itself on top of the manifest with all other module identifiers at the end of the manifest (e.g. Fig. 3B), but **fails to disclose** providing a hash of the contents of such assembly at the end of the assembly. But this limitation to provide hash of the contents of modules has been addressed in claim 1 above. Hence it would have been obvious for one of ordinary skill in the art at the time the invention was made to add such hash of contents as taught by Buxton at the end of the assembly to enhance the lay out of the manifest in order to expedite information and facilitate efficiently the integrity checking as intended per module when resources at built time are available as mentioned in claim 1 above.

As per claim 6, Renaud in combination with Buxton, discloses version number included in the manifest (Fig. 3b) and security checking of modules and comparing of signatures identifier from the manifest and generated from the module (Fig. 4, 6); but does expressly mention determining if the contents of the assembly has been modified by comparing the actual hash from the module contents with the hash stored in the manifest. Buxton, similar to Renaud, discloses if data module has not been tampered with when authenticating it against the signature of the publisher; further, Buxton teaches integrity checking to see if data is corrupted using licensing and registration mechanisms (e.g. *check license and integrity, component has been damaged* - col. 17, lines 16-44); hence suggests whether the loaded module has been the same as

originally created and registered. Buxton's way of comparing to determine if the module contents as hashed have been modified is therefore implicitly disclosed. In case Renaud's hashed signature comparing does not already include a verification as to whether a module has been modified, it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide a integrity checking such as taught by Buxton to the data authentication by Renaud's using the version number in the manifest as mentioned above to determine if the up/down-loaded module for activation are integral with respect to their predetermined version, thus enhancing further the integrity checking of data and security control as intended by both Renaud and Buxton.

As per claims 7 and 8, Renaud discloses version and publisher information (e.g. col. 6, lines 55-64) and thereby suggests which version determination but fails to expressly disclose if the assembly has been modified using such information (re claim 7) but this limitation has been addressed in claim 6 above. Further, Renaud does disclose whether the publisher of the assembly is (re claim 7) trustworthy (e.g. step 506 – Fig. 5, 5a; col. 9, lines 8-21) via authenticating publisher name information in the manifest(re claim 8).

As per claim 18, Renaud discloses a computer readable medium (col. 15, lines 13-30) with an executable for a runtime application program, such medium comprising

an assembly (e.g. *data structure 300* – Fig. 3a) including manifest containing a list of modules making up the assembly (e.g. Fig. 3A); and

a hash of at least one list of modules in the assembly (e.g. identifier *316, 318* - Fig. 3B; col. 7, lines 15-27).

But Renaud does not specify that a hash of the list of module in the manifest is a hash of the contents of at least one module. But this limitation has been addressed above in claim 1 using Buxton's teachings.

As per claim 23, this is a system claim version of claim 1 above; hence is rejected herein using the corresponding rejection set forth therein.

As per claim 24, this claim recites comparing hash of a module in the manifest and hash of actual module is analogous to the comparison limitation as recited in claim 6 above. Hence, the rejection in claim 6 herein applies.

As per claim 25, Renaud discloses identity, publisher and version information and Buxton discloses integrity checking of code assemblies or modules as mentioned in claims 3-4 and 6-8 when addressing trustworthiness of hash value, publisher and version information above. Based on such teachings and rationale of rejection as set forth therein, the limitation of using identity and version information to determine if the assembly should be executed would also have been obvious using the rationale of mainly claims 6-8.

7. Claims 10-13, and 27-35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Renaud et al., USPN: 5,958,051, in view of Evans et al., USPN: 5,805,899 (hereinafter Evans).

As per claim 10, Renaud discloses a method for facilitating integrity checking employable by application programs at runtime (e.g. Fig. 6; col. 9, line 40 to col. 10, line 37), such method comprising:

providing an assembly (e.g. *data structure 300* – Fig. 3a) with manifest (Fig. 3b) that contains a list of assemblies (e.g. *data files 304-314, version of the file* - col. 6, lines 46-64; col.

7, lines 31-36 – Note: data files, digital stream or class files are assemblies of other sub-assemblies like subclasses or subdata) that make up the assembly;

providing a manifest (*signature file 302* – Fig. 3b)with a hash of the contents of at least one assemblies (e.g. *identifier, one-way hash* - col. 7, lines 15-27).

But Renaud does not discloses that the manifest contains a list of referenced assemblies that the assembly depends on; nor does Renaud disclose a manifest with a hash of a manifest of one referenced assembly of the list of referenced assemblies. However, Renaud teaches representing a manifest with a hashed representation (*signature 322* - Fig. 3B) and suggests dependency on multiple location (hashed) identifiers by one assembly at verification and runtime (e.g. col. 9, lines 30-39; *Add site* - Fig. 5a; col. 4, lines 9-34; Fig. 18). In a method to bind objects at runtime using hash representation of versioned assemblies analogous to the Renaud's signature file listing, Evans discloses pointer means to refer to other manifest of other assemblies that the pointing assembly depends on (re claim 9). Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to modify the representation of manifest by Renaud with the implementation suggested by Evans to point to other assemblies referenced to by the pointing assembly in providing a manifest with the list of referenced assemblies (or files or objects) and using therein pointer information to point to the hash representation of the manifest of the assembly referred to by such pointer because of the same benefits as set forth in claim 9 above.

As for claim 11, the limitation would also have been obvious in view of the rationale as set forth in claim 10; because providing in a manifest a hash of each of the referenced assemblies

would enable more time-efficient locating of depended-upon assemblies as well as verifying their integrity and authentication as intended by Renaud.

As per claims 12 and 13, Renaud also teaches identity information in the manifest (re claim 3) and publisher and version information (re claim 4).

As per claim 27, Renaud discloses a method for facilitating integrity of assemblies employable by application program at runtime, such method comprising components for:

a first component to provide a manifest of assembly (e.g. *signature file 302* – Fig. 3b; *data structure 300* – Fig. 3a - Note : data structure 300 is assembly and signature file is manifest) that contains at least one a sub-assemblies (e.g. *data files 304-314, version of the file* - col. 6, lines 46-64; col. 7, lines 31-36 – Note: files, digital stream or class files are sub-assemblies making up assembly structure 300) that make up the assembly, such sub-assemblies comprise a manifest (e.g. Fig. a-b – Note: file representation by identifier 316, 318 is equivalent to manifest of each sub-assemblies); and

a second component to provide the manifest with a hash of the sub-assemblies (e.g. *identifier, one-way hash* - col. 7, lines 15-27 – Note: hash representation of file identifier is hash of the manifest of the sub-assemblies).

But Renaud does not discloses that the sub-assembly in the manifest is a referenced assembly; but this limitation has been addressed using the teachings by Evans to link assemblies to other assemblies reference using the rationale as set forth in claim 10 above.

Nor does Renaud disclose that such referenced assembly comprises a manifest. But Evans discloses multiple representation of referenced assemblies by a structure with manifest

depiction of assembly identification information, i.e. each referenced assembly in turn including manifest information as to represent itself (e.g. structure # - Fig. 11).

Nor does Renaud disclose that the assembly manifest includes a hash of the manifest of at least one of the referenced assembly. However, Evans discloses representing each referenced assembly with a manifest including a hash of the referenced assembly and Renaud discloses hash of each assembly manifest (*signature file 302* – Fig. 3b).

Based on the teachings by Evans to provide hash information on each referenced assembly referred to by the main assembly and by Renaud to make a hash for each assembly manifest from above, the motivation to provide the assembly manifest with referenced assemblies having each a manifest and to provide the assembly manifest with a hash of such referenced assembly manifest would have been obvious for the same reasons as set forth in claim 10 above.

As per claim 28, the comparing of hash value limitation is rejected using the same rejection as set forth in claim 6 above because of the similarity in intent and purpose.

As per claim 29, Renaud does not teach a binding policy but discloses establishing and verification of signature with version information included and access permissions protocol (e.g. Fig. 5, 5a, 7) to enable trusted communication to bind usage of received files to the secure and trusted protocols; and also teaches determining which applet is to be trusted for execution (Fig. 6). Evans discloses binding assemblies to another version representation of another assembly (Evans: Fig. 11). It would have been obvious for one of ordinary skill in the art at the time the invention was made to modify Renaud's method for integrity checking using trusted protocol of versioned assemblies such as to enhance it with the version dependency linking and correct

version determining policy by Evans and effect it such to determine which version is to be executed when another version is resident on the executing environment because this would enable alleviating system errors from having uncontrolled versions dynamically competing in a same system.

As per claim 30, the limitation of related assembly is equivalent to referenced assembly as recited in claim 27 above, hence this claim is rejected with the same rejection as set forth in claim 27 above.

As per claim 31, Renaud teach about module (re claim 1) and Evans teaches about related assembly. The motivation to make such manifest of related assembly to represent the module of Renaud would have been obvious in light of the rationale in claim 27 above.

As per claims 32-35, refer to claims 27, 28, 29, and 21 respectively.

8. Claims 14-17, and 22 are rejected under 35 U.S.C. 103(a) as being unpatentable over Renaud et al., USPN: 5,958,051, and Evans et al., USPN: 5,805,899, as applied to claims 10 and 16 (for 14-17) and further in view of Buxton, USPN: 6,182,279.

As per claim 14, Renaud discloses hash of the identifiers of versioned modules in the assembly and suggests placing a signature (or hashed representation) of the manifest itself on top of the manifest with all other module identifiers at the end of the manifest (e.g. Fig. 3B), but **fails to disclose** providing a hash of the contents of such assembly at the end of the assembly. But this limitation to provide hash of the contents of modules has been addressed in claim 1 above, using Buxton's teaching. Hence it would have been obvious for one of ordinary skill in the art at the time the invention was made to add such hash of contents as taught by Buxton at the end of the assembly to enhance the lay out of the manifest in order to expedite information and facilitate

efficiently the integrity checking as intended per module when resources at built time are available as mentioned in claim 1 above.

As per claim 15, the limitation as to determine whether the contents of the referenced assembly has been modified would also have been obvious in light of the rationale set forth in claim 6 which is further applied to the combination of Renaud/Evans of claim 10 because of the same benefits as set forth in claim 6.

As per claims 16 and 17, the rationales based on Buxton's teachings and used in the rejections of claims 7, 8, are herein respectively applied to the combination of Renaud/Evans for the same motivation as set forth therein correspondingly.

As per claim 22, Renaud discloses a readable medium (re claim 18) such medium comprising an assembly with a manifest containing a list of referenced assemblies. But such limitations have been addressed in claim 10 and 14 above; using the corresponding rationale as set forth therein to address the referenced assemblies limitation and hash of contents of referenced assemblies.

9. Claims 9, 19-21, 26 are rejected under 35 U.S.C. 103(a) as being unpatentable over Renaud et al., USPN: 5,958,051, and Buxton, USPN: 6,182,279, as applied to claim 1, 18, 23, and further in view of Evans et al., USPN: 5,805,899.

As per claim 9, in reference to claim 1, Renaud discloses a signature or hashed representation of the manifest file (Fig. 3B) and identification of multiple sites signature (hashed representation) for a given loaded files (col. 9, lines 30-39; *Add site* - Fig. 5a) to verify the authenticity of the module in the loaded assembly (e.g. col. 4, lines 9-34) of another versioned object (Fig. 18) **but does not disclose** a manifest providing hash of another manifest

upon which it depends on. Buxton teaches signature of components (re claim 1) and pointer information to control dependency of components in the aggregate of OLE components to combine at runtime template (e.g. col. 9, lines 45-67). Evans, in a method to link runtime versioned objects similar to the combination of components of Buxton's template using hash value of object identifier (Evans: Fig. 11) analogous to that such as taught by Renaud (Renaud: Fig. 4), discloses pointer information to refer to the assembly manifest of another versioned object that the assembly depends on (e.g. Fig. 16). In view of the pointer reference teachings by both Buxton and Evans, combined with the hash representation of the manifest by Renaud, it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide the manifest file or signature file of Renaud, with pointer information to locate the site of another manifest site or assembly that the pointing assembly depends on, such location represented by a hashed identifier as suggested by Renaud. One of ordinary skill in the art would be motivated to do so because this would extend the security/version checking adopted by Renaud in that it reduces time to locate a referred to site of another assembly of modules, such reduction of time being enhanced by using pointing means as suggested by Buxton and furthered by Evans, to locate and verify the hashed representation of the depended-upon assembly, or hash of the manifest of the assembly pointed to by the dependent assembly.

As per claim 19, this claim includes a manifest with a list of referenced assembly and a hash of manifest of such referenced assembly, just as recited in claim 10, hence will be rejected herein with the rationale of claim 10 using the combined teachings of Renaud/Buxton and the referenced assemblies teaching by Evans.

As per claim 20, refer to rejection of claim 4 or 13.

As per claim 21, Renaud mentions about application classes used in a GUI window environment (e.g. Fig. 5a) to provide interactive verification of applets and Buxton discloses template storage DLL to enhance the component builder (col. 7, lines 48-61). Official notice is taken that the use of Dynamic Linked Library to effect windows application and user interface functionality was a well-known concept at the time of the invention. In case the components to build by Buxton or the assembly of code or modules by Renaud do not already include a Dynamic Linked Library (DLL), it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide a DLL as one type of assembly of code in the list of assemblies as taught by Renaud/Buxton, because of the known benefits ascribed to using DLL such as portability, storage benefits and ease to use without the need for recompilation.

As per claim 26, Renaud does not expressly teach a binding policy but discloses establishing and verification of signature with version information included and access permissions protocol (e.g. Fig. 5, 5a, 7) to enable trusted communication to bind usage of received files to the secure and trusted protocols; and also teaches determining which applet is to be trusted for execution (Fig. 6). In building components using secure object verification means, Buxton further enhances Renaud's version recording by disclosing integrity checking to verify if data are not corrupted or modified (*check license and integrity, component has been damaged* - col. 17, lines 16-44). And Evans discloses binding assemblies to another version representation of another assembly and ensuring that the correct version of object gets executed (Evans: Fig. 11, 15). It would have been obvious for one of ordinary skill in the art at the time the invention was made to modify Renaud's (enhanced by Buxton) method for integrity checking using trusted protocol of versioned assemblies such as to enhance it with the version dependency linking and

correct version determining policy by Evans and effect it such to determine which version is to be executed when another version is resident on the executing environment because this would enable alleviating fatal system errors from having uncontrolled versions dynamically competing in a same system.

Conclusion

10. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Pat No. 6,510,516 to Benson et al., disclosing plurality of hash representation of each sub-components loaded.

U.S. Pat No. 6,480,880 to White, disclosing jar file with hash index for each loaded class.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 872-9306 (for formal communications intended for entry)

or: (703) 746-8734 (for informal or draft communications, please label "PROPOSED" or "DRAFT" – please consult Examiner before use)

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA. , 22202. 4th Floor(Receptionist).

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

VAT
December 8, 20033

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100